

Serial No: 09/854,031

IN THE CLAIMS:

Please amend claims 7, 21, 50, 60, 61, 71, 81, 82, 87 and 88 as follows:

1 Claim 1. (original) A method comprising:
2 employing a computer for:
3 obtaining a collection of code;
4 providing a program graph representing said collection of code;
5 identifying any authorization resources of said collection of code;
6 locating any bounded path within said program graph; and
7 associating said any authorization resource with said any bounded path.

1 Claim 2. (original) A method as recited in claim 1, wherein said
2 collection of code includes codes obtained from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 3. (original) A method as recited in claim 1, wherein the step
2 of providing includes constructing said program graph through static analysis
3 techniques.

1 Claim 4. (original) A method as recited in claim 3, wherein the step
2 of constructing includes employing object code.

1 Claim 5. (original) A method as recited in claim 1, wherein the step
2 of identifying includes finding at least one authorization point in said
3 program graph.

1 Claim 6. (original) A method as recited in claim 5, wherein the step
2 of finding includes finding a Java authorization point.

1 Claim 7. (currently amended) A method as recited in claim 6, wherein
2 the step of ~~find~~ finding the a Java authorization point includes finding a
3 AccessController.checkPermission node in said program graph, and finding a
4 java.security.Permission object passed as an argument to the
5 AccessController.checkPermission method.

1 Claim 8. (original) A method as recited in claim 5, wherein said
2 authorization point is an instruction invocation.

1 Claim 9. (original) A method as recited in claim 8, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

Serial No: 09/854,031

1 Claim 10. (original) A method as recited in claim 9, wherein said
2 particular language is C#.

1 Claim 11. (original) A method as recited in claim 1, wherein the step
2 of identifying includes employing data flow analysis.

1 Claim 12. (original) A method as recited in claim 11, wherein the step
2 of employing includes generating a data flow from said program graph.

1 Claim 13. (original) A method as recited in claim 1, wherein the step
2 of identifying any bounded path includes locating a set of start nodes in
3 said program graph, and locating a stop node in said program graph; and said
4 bounded path includes all nodes within the graph bound by said start nodes
5 and said stop node.

1 Claim 14. (original) A method as recited in claim 1, wherein the step
2 of associating includes associating and aggregating said any authorization
3 resource with said collection of code.

1 Claim 15. (original) An apparatus comprising:
2 computing means having:
3 means for obtaining a collection of code;
4 means for providing a program graph representing said collection of
5 code;
6 means for identifying any authorization resources of said collection of
7 code;
8 means for locating any bounded path within said program graph; and
9 means for associating said any authorization resource with said any
10 bounded path.

1 Claim 16. (original) An apparatus as recited in claim 15, wherein said
2 collection of code includes codes obtained from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 17. (original) An apparatus as recited in claim 15, wherein the
2 means for providing includes means for constructing said program graph
3 through static analysis techniques.

1 Claim 18. (original) An apparatus as recited in claim 17, wherein the
2 means for constructing includes employing object code.

Serial No: 09/854,031

1 Claim 19. (original) An apparatus as recited in claim 15, wherein the
2 means for identifying includes means for finding at least one authorization
3 point in said program graph.

1 Claim 20. (original) An apparatus as recited in claim 19, wherein the
2 means for finding includes finding a Java authorization point.

1 Claim 21. (currently amended) An apparatus as recited in claim 20,
2 wherein the means for ~~find~~ finding a Java authorization point includes
3 finding a AccessController.checkPermission node in said program graph, and
4 finding a java.security.Permission object passed as an argument to the
5 AccessController.checkPermission method.

1 Claim 22. (original) An apparatus as recited in claim 19, wherein said
2 authorization point is an instruction invocation.

1 Claim 23. (original) An apparatus as recited in claim 22, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

1 Claim 24. (original) An apparatus as recited in claim 23, wherein said
2 particular language is C#.

1 Claim 25. (original) An apparatus as recited in claim 15, wherein the
2 means for identifying includes employing data flow analysis.

1 Claim 26. (original) An apparatus as recited in claim 25, wherein the
2 means for employing includes generating a data flow from said program graph.

1 Claim 27. (original) An apparatus as recited in claim 15, wherein the
2 means for identifying any bounded path includes locating a set of start nodes
3 in said program graph, and locating a stop node in said program graph; and
4 said bounded path includes all nodes within the graph bound by said start
5 nodes and said stop node.

1 Claim 28. (original) An apparatus as recited in claim 15, wherein the
2 means for associating includes associating and aggregating said any
3 authorization resource with said collection of code.

1 Claim 29. (original) A method comprising:
2 employing a computer including the steps of:
3 obtaining a collection of code;
4 providing a program graph representing said collection of code; and

Serial No: 09/854,031

5 identifying a complete set of authorization resources of said
6 collection of code.

1 Claim 30. (original) A method as recited in claim 29, if no resource
2 is identified in said step of identifying, further comprising providing an
3 indication that authorization testing is not necessary.

1 Claim 31. (original) A method as recited in claim 29, further
2 comprising:
3 identifying any bounded path within said program graph; and
4 associating any authorization resource identified in the step of
5 identifying with said any bounded path.

1 Claim 32. (original) An apparatus comprising:
2 a computer having access to a collection of code, the computer
3 including:
4 an authorization resource identifier to identify any authorization
5 resources within the collection of code;
6 a bounded path locator to locate any bounded path within a program
7 graph of said collection of code; and
8 an associator to associate said any authorization resource with said
9 any bounded path.

1 Claim 33. (original) An apparatus as recited in claim 32, wherein said
2 collection of code includes codes obtained from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 34. (original) An apparatus as recited in claim 32, further
2 comprising a program graph constructor to construct said program graph
3 through static analysis techniques.

1 Claim 35. (original) An apparatus as recited in claim 32, wherein the
2 program graph constructor uses object code of said collection of code.

1 Claim 36. (original) An apparatus as recited in claim 32, wherein the
2 authorization resource identifier finds at least one authorization point in
3 said program graph.

1 Claim 37. (original) An apparatus as recited in claim 36, wherein at
2 least one authorization point of said at least one authorization point is a
3 Java authorization point.

1 Claim 38. (original) An apparatus as recited in claim 37, wherein the
2 Java authorization point includes finding a AccessController.checkPermission

Serial No: 09/854,031

3 node in said program graph, and finding a java.security.Permission object
4 passed as an argument to the AccessController.checkPermission method.

1 Claim 39. (original) An apparatus as recited in claim 36, wherein said
2 authorization point is an instruction invocation.

1 Claim 40. (original) An apparatus as recited in claim 39, wherein said
2 instruction invocation is used in a particular language for said collection
3 of code.

1 Claim 41. (original) An apparatus as recited in claim 40, wherein said
2 particular language is C#.

1 Claim 42. (original) An apparatus as recited in claim 32, wherein the
2 authorization resource identifier employs data flow analysis.

1 Claim 43. (original) An apparatus as recited in claim 42, wherein the
2 data flow analysis is generated from said program graph.

1 Claim 44. (original) An apparatus comprising:
2 a computing module having access to a collection of code, the computer
3 module including:
4 an authorization resource identifier to completely identify any
5 authorization resources within a collection of code; and
6 a bounded path locator to locate any bounded path within a program
7 graph of said collection of code.

1 Claim 45. (original) An apparatus as recited in claim 44, wherein the
2 authorization resource identifier provides an indication that authorization
3 testing is not necessary if no resource is identified by said authorization
4 resource identifier.

1 Claim 46. (original) An apparatus as recited in claim 44, further
2 comprising an associator to associate said any authorization resource with
3 said any bounded path.

1 Claim 47. (original) An apparatus comprising:
2 a computer including:
3 means for obtaining a collection of code;
4 means for providing a program graph representing said collection of
5 code; and
6 means for identifying a complete set of authorization resources of said
7 collection of code.

Serial No: 09/854,031

1 Claim 48. (original) An apparatus as recited in claim 47, further
2 comprising means for indicating if authorization testing is necessary.

1 Claim 49. (original) An apparatus as recited in claim 47, further
2 comprising:
3 means for identifying any bounded path within said program graph; and
4 means for associating any authorization resource identified in the step
5 of identifying with said any bounded path.

1 Claim 50. (currently amended) A method comprising:
2 employing a computer for:
3 constructing a program graph from a collection of code using static
4 analysis techniques;
5 performing a data flow analysis using static analysis techniques;
6 searching said program graph for any ~~useful~~ resource for executing said
7 collection of code;
8 identifying any bounded path within said program graph over which said
9 ~~useful~~ resource is ~~beneficial~~ utilized; and
10 associating said ~~useful~~ resource with said collection of code.

1 Claim 51. (original) The method as recited in claim 50, wherein the
2 step of constructing includes employing source code of said collection of
3 code.

1 Claim 52. (original) A method as recited in claim 50, wherein the step
2 of constructing includes building an invocation graph of said collection of
3 code to form said program graph.

1 Claim 53. (original) A method as recited in claim 50, wherein the step
2 of constructing a program graph includes constructing a call graph of said
3 collection of code to form said program graph.

1 Claim 54. (original) A method as recited in claim 50, wherein said
2 step of constructing includes using context-sensitivity.

1 Claim 55. (original) A method as recited in claim 54, wherein said
2 step of using context sensitivity includes using type information for any
3 method receiver and/or any parameter.

1 Claim 56. (original) A method as recited in claim 55, wherein the step
2 of using type information includes using class and memory allocation site
3 information.

Serial No: 09/854,031

1 Claim 57. (original) A method as recited in claim 56, wherein the step
2 of using type information includes using per instance information.

1 Claim 58. (original) A method as recited in claim 57, wherein the step
2 of using instance information includes associating instance information with
3 a node or edge in said program graph.

1 Claim 59. (original) A method as recited in claim 50, wherein said
2 collection of code includes codes constructed from a group of codes including
3 basic blocks, class methods, classes, collections of classes or any
4 combination of these.

1 Claim 60. (currently amended) A method as recited in claim 50, wherein
2 the step of searching includes locating a node or edge in said program graph
3 that represents a location where said ~~useful~~ resource would be ~~beneficial~~
4 utilized.

1 Claim 61. (currently amended) A method as recited in claim 60, wherein
2 said ~~useful~~ resource is a resource identifier.

1 Claim 62. (original) A method as recited in claim 60, wherein said
2 location is an authorization test.

1 Claim 63. (original) A method as recited in claim 50, wherein said
2 program graph represents an object oriented program.

1 Claim 64. (original) A method as recited in claim 50, wherein said
2 program graph and said data flow analysis are constructed from Java object
3 code.

1 Claim 65. (original) A method as recited in claim 61, wherein said
2 resource identifier includes at least one Java java.security.Permission.

1 Claim 66. (original) A method as recited in claim 62, wherein said
2 authorization test is a call to any
3 java.security.AccessController.checkPermission method.

1 Claim 67. (original) A method as recited in claim 62, wherein said
2 location represents a call to any authorization testing method in any
3 instance of java.lang.SecurityManager and/or one of its subclasses.

1 Claim 68. (original) A method as recited in claim 55, wherein said
2 node has a parameter which said type information is a Java
3 java.security.Permission.

Serial No: 09/854,031

1 Claim 69. (original) A method as recited in claim 68, wherein the step
2 of identifying includes locating the constructor for said Java
3 java.security.Permission allocation site and using said data flow analysis in
4 identifying any value passed by any parameter to said constructor, wherein
5 the combination of the Java java.security.Permission and a value for any
6 parameter is the used Permission.

1 Claim 70. (original) A method as recited in claim 50, wherein the step
2 of identifying any bounded path includes locating a set of start nodes in
3 said program graph, and locating a stop node in said program graph; and said
4 bounded path includes all nodes within the graph bound by said start nodes
5 and said stop node.

1 Claim 71. (currently amended) A method as recited in claim 50, wherein
2 the step of associating includes mapping a subset of said collection of code
3 to said ~~useful~~ resource.

1 Claim 72. (original) A method as recited in claim 71, wherein the
2 subset of said collection of code includes nodes in said bounded path.

1 Claim 73. (original) A method as recited in claim 72, further
2 comprising employing a privileged Java code wherein said stop node represents
3 the method java.security.AccessController.checkPermission(); and
4 employing said start nodes are any of root nodes in said program graph
5 or a node representing the method
6 java.security.AccessController.doPrivileged().

1 Claim 74. (original) A method as recited in claim 73, further
2 comprising connecting a used Permission with said privileged Java code.

1 Claim 75. (original) A method as recited in claim 74, further
2 comprising connecting a used Permission with any node in said program graph
3 prior to said java.security.AccessController.doPrivileged() node.

1 Claim 76. (original) A method as recited in claim 74, wherein said
2 step of associating includes connecting a used Permission for each
3 java.security.AccessController.checkPermission() in said program graph.

1 Claim 77. (original) A method as recited in claim 76, wherein said
2 step of associating includes connecting said used Permission from each node
3 in said program graph to each method associated with said program graph.

Serial No: 09/854,031

1 Claim 78. (original) A method as recited in claim 77, wherein said
2 step of associating includes connecting said used Permission from each method
3 in said program graph to each class in said program graph.

1 Claim 79. (original) A method as recited in claim 78, wherein said
2 step of associating includes connecting said used Permission from each class
3 in said program graph to collection of classes.

1 Claim 80. (original) A method as recited in claim 79, wherein said
2 step of associating includes connecting said used Permission is associated
3 from each class in said program graph to collection of classes.

1 Claim 81. (currently amended) A method as recited in claim 50, further
2 comprising employing said ~~useful~~ resource in executing said collection of
3 code.

1 Claim 82. (currently amended) A method comprising:
2 statically detecting ~~useful~~ resources for a collection of code written
3 in a computer programming language, by including the steps of:
4 calculating if any code of said collection of code is part of a program
5 graph;
6 identifying any ~~useful~~ resource for said collection of code;
7 determining any bounded path of nodes within said program graph which
8 constrain said ~~useful~~ resources of said collection of code; and
9 associating said any ~~useful~~ resource with said collection of code
10 within said bounded path of nodes in said program graph.

1 Claim 83. (original) A method as recited in claim 82, wherein the step
2 of calculating includes using an invocation graph having a set of root nodes
3 for said program graph.

1 Claim 84. (original) A method as recited in claim 82, wherein the step
2 of calculating includes using a call graph having a set of root nodes for
3 said program graph.

1 Claim 85. (original) A method as recited in claim 83, further
2 comprising determining whether a basic block, method, class is represented as
3 a node in said invocation graph.

1 Claim 86. (original) A method as recited in claim 84, further
2 comprising determining whether a basic block, method, class is represented as
3 a node in said call graph.

Serial No: 09/854,031

1 Claim 87. (currently amended) A method as recited in claim 83, further
2 comprising identifying a node in said invocation graph that identifies said
3 ~~useful~~ resources, wherein said ~~useful~~ resources is one or more resources.

1 Claim 88. (currently amended) A method as recited in claim 84, further
2 comprising a step identifying a node in said invocation graph that identifies
3 said ~~useful~~ resources, wherein said ~~useful~~ resources is one or more
4 resources.

1 Claim 89. (original) A method as recited in claim 83, wherein nodes in
2 said bounded path of nodes are all nodes in the graph that are bounded
3 between start nodes and a stop node.

1 Claim 90. (original) A method as recited in claim 84, wherein nodes in
2 said bounded path of nodes are all nodes in the graph that are bounded
3 between a set of start nodes and a stop node.

1 Claim 91. (original) A method as recited in claim 90, wherein all root
2 nodes in said invocation graph are members of the start nodes.

1 Claim 92. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 1.

1 Claim 93. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 29.

1 Claim 94. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 article of manufacture comprising computer readable program code means for
5 causing a computer to effect the steps of claim 50.

1 Claim 95. (original) An article of manufacture comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing statistical detection, the computer readable program code means
4 in said article of manufacture comprising computer readable program code
5 means for causing a computer to effect the steps of claim 80.

Serial No: 09/854,031

1 Claim 96. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 15.

1 Claim 97. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing association, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 32.

1 Claim 98. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 44.

1 Claim 99. (original) A computer program product comprising a computer
2 usable medium having computer readable program code means embodied therein
3 for causing identification, the computer readable program code means in said
4 computer program product comprising computer readable program code means for
5 causing a computer to effect the functions of the elements in claim 47.

1 Claim 100. (original) A program storage device readable by machine,
2 tangibly embodying a program of instructions executable by the machine to
3 perform method steps for identification, said method steps comprising the
4 steps of claim 1.